The implementation described above uses node-link data that include expansion flags of links to define a tree within a graph as disclosed in copending coassigned U.S. Patent Application ~~09/EEE,EEE~~ 09/124,338 (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the Graph", with memory management as disclosed in copending coassigned U.S. Patent Application ~~09/DDD,DDD~~ 09/124,474 (Attorney Docket No. D/98205Q3), entitled "Controlling Which Part of Data Defining a Node-Link Structure is in Memory", both incorporated herein by reference, but the invention could be implemented with a node-link structure defined in any other appropriate way, and loaded into memory in any appropriate way.

Change(s) applied to document, /J.L.B./ 1/6/2012

The implementation described above employs a directed graph data structure in which a link is represented as an item in two linked lists, one for the outgoing links from its from-node and one for the incoming links to its to-node. Any other suitable data structure could be employed.

The implementation described above can handle directed graphs, including cyclic directed graphs, but the invention could be implemented for other types of graphs by converting other types of links to appropriate combinations of directed links or by otherwise providing a protocol for mapping the structure of a graph to a tree. For example, an undirected link between two nodes could be converted to a pair of directed links between the same nodes or could be assigned a direction based on an appropriate criterion. In general, a representation in which all undirected links have been converted to a pair of

by obtaining a layout with the least overlap in areas. Layout results could be cached to minimize repetitive computation.

In the implementation described above, a node-link structure laid out in the hyperbolic plane is then mapped into the unit disk and then painted in accordance with copending coassigned U.S. Patent Application 09/124,529 ~~09/CCC,CCC~~ (Attorney Docket No. D/98205Q2), entitled "Mapping a Node-Link Structure to a Rendering Space Beginning from any Node", incorporated herein by reference, but a node-link structure laid out in accordance with the invention could be laid out in any other appropriate negatively curved space, and then be handled in any other appropriate way, with or without mapping, or mapped and presented in any other appropriate way, including mapping it into any other appropriate rendering space and presenting it in any other appropriate display space, including three-dimensional rendering and display spaces.

Change(s) applied to document, /J.L.B./ 1/6/2012

The implementation described above determines whether to lay out a node's children by comparing an angle displacement obtained for the node with a previous angle displacement, but the invention could be implemented by laying out all descendants of each laid out node or by applying any other suitable criterion to determine which elements to lay out.

The implementation described above is suitable for laying out elements of a tree. The invention could be used to lay out elements of other types of node-link structures, such as graphs in general.

node-link structure like the representations disclosed in Lamping et al., US-A-5,619,632 and in copending coassigned U.S. Patent Application ~~09/AAA,AAA~~ 09/124,528 (Attorney Docket No. D/98205), entitled "Presenting Node-Link Structures with Modification", both incorporated herein by reference. The invention could, however, be implemented with or without navigation signals; for example, elements could be moved around in response to different sortings of the children of a node or in response to the application of different filters to elements of a structure. Also, the invention could be implemented with any appropriate type of expand and contract signals or other navigation signals, including signals resulting from external queries, selection of a menu entry-like item requesting expansion below an indicated node or link, or selection of a menu entry-like item requesting expansion below the current focus. The navigation signals could instead relate to an illusory space like those produced by videogames or virtual reality environments or a presentation space other than a display and navigation signals could instead be produced by any appropriate user input device, including other kinds of pointing devices and other kinds of devices for receiving alphanumeric or linguistic input such as voice, gestures, or other modes of user input. Further, the invention could be implemented with other types of representations of node-link structures. The invention could be implemented without animation or with any appropriate animation techniques.

Change(s) applied to document, /J.L.B./ 1/6/2012

The implementation described above obtains nearby relationship data relating to relationships among a parent, its children, and its grandchildren, but

root node at the new orientation before mapping and painting, in box 404. The root node can be laid out as described above in relation to box 352 in Fig. 6, but with the new orientation. The new orientation will then be used in mapping, changing the orientation of the representation.

If the change is a non-animated edit, as could occur in response to a stretch event, a drag event, a bookmark event, or a click event if edits are pending, walker routines 222 first set up a list of remove edits, in box 410, and then lay out the remove edits before mapping and painting, in box 412. Then, walker routines 222 set up a list of add edits, in box 414, and then lay out the add edits before mapping and painting, in box 416.

In the current implementation, the lists of edits are set up based on edit source lists that are maintained by various routines in memory 214, including grapher routines 220 and painter routines 224. Also, the current implementation relates to a tree defined by expanded links, as explained in copending coassigned U.S. Patent Application ~~09/EEE,EEE~~ 09/124,338 (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the Graph", incorporated herein by reference. One pair of edit source lists, designated "CollapsedLinks" and "ExpandedLinks" herein, includes edits for links selected by contract requests and expand requests, respectively, and can therefore be set up in box 330 in Fig 5. The other pair, designated "RemovedLinks" and "AddedLinks" herein, includes edits for links that are

Change(s) applied to document, /J.L.B./ 1/6/2012

09/~~EEE,EEE~~ (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the Graph", incorporated herein by reference.

The test in box 380 applies an appropriate criterion to determine whether to continue layout to the next generation of nodes. As noted above in relation to box 374, the criterion can be whether any child node's angle has been modified by more than a small angular difference, such as 0.00001. If so, layout should continue.

In box 382, walker routines 222 push the ID of each child node that is expanded or not a leaf onto the front of the queue. Other child nodes could be marked walked in box 382, since they do not have children that will be laid out. When box 382 is completed or if the test in box 380 determines not to continue or the test in box 370 determines that the node is already walked, the back node on the queue is popped, in box 384, before returning to box 360.

Fig. 7 illustrates how layout of a changed node-link structure can be performed in boxes 312, 324, and 334 in Fig. 5. In each case, layout begins in response to a call that leads to layout and mapping, as shown in box 400. As illustrated by the branch in box 402, however, the manner in which layout is performed depends on the type of change being made in the node-link structure.

If the change is a change in orientation of the root node in response to an orientation event, walker routines 222 can call math routines 226 to lay out the

U. S. Patent 5,590,250, incorporated herein by reference. This function, referred to herein as "RoomAvailable", starts with a distance D that has been moved into a wedge and an angle Φ that is half of the wedge. RoomAvailable returns a distance to the edge of the wedge that is calculated by first obtaining the ratio $(1-D^2)/2D$, and by then dividing the ratio by $\sin\Phi$ to obtain an initial distance S. RoomsAvailable then returns the distance $((S^2-1)^{1/2}-S)$. To obtain a child's area, RoomAvailable is called with the same distance and angle that were used in calling InsideAngle, as described above. The distance returned by RoomAvailable is saved as a measure of the child's area.

Although the software implementation described above can save additional data, it is based on the discovery that only two items of data need to be stored for each node in order to be able to perform layout and mapping as described herein and in relation to Fig. 6 of copending coassigned U.S. Patent Application ~~09/CCC,CCC~~ 09/124,529 (Attorney Docket No. D/98205Q2), entitled "Mapping a Node-Link Structure to a Rendering Space Beginning from any Node", incorporated herein by reference. One item indicates a distance or position displacement from the node to its children nodes in the hyperbolic plane. The other is an angle displacement in the hyperbolic plane between the extension of the incoming link to the node's parent and outgoing link from the parent to the node. These two items of data, or a handle that can be used to access them, can be included in a link's data item in a directed graph data structure as illustrated in Fig. 6 of copending coassigned U.S. Patent Application

Change(s) applied to document, /J.L.D./ 1/6/2012

deletions and insertions. The technique has been successfully implemented to produce object constancy during these movements.

After a representation is provided in box 316 or after an animation sequence is completed in box 322 or 332, another event can be received in box 302, as indicated by the circles labeled "A" in Fig. 5.

Animation details relating to the loops that begin in boxes 322 and 332 are discussed in more detail in copending coassigned U.S. Patent Application 09/124,528 ~~09/AAA,AAA~~ (Attorney Docket No. D/98205), entitled "Presenting Node-Link Structures with Modification", incorporated herein by reference.

Change(s) applied to document, /J.L.B./ 1/6/2012

C.3. Layout

Fig. 6 shows how layout can be initially performed in box 300 in Fig. 5. Fig. 7 shows how layout of a changed node-link structure can be performed in boxes 312, 324, and 334.

As shown in box 350, walker routines 222 begin initial layout by obtaining the root node ID and using it to access data relating to the root node in directed graph data structure 232. In box 352, walker routines 222 lay out the root node by making a call to math routines 226 with an angle width. This could be any suitable angle that produces a desirable result. The angles 2π and $\pi/2$ have been successfully used, with 2π appropriate for a center layout style and with $\pi/2$ appropriate for a top, bottom, right, or left layout style. An interface could also be provided to modify this angle to obtain desirable results.

a swap of display buffers can be performed to present the tree as painted, thus providing a representation of the graph.

When a new edit is marked in box 328 by painter routines 224, the new edit is laid out during the next iteration, in box 324. As a result, the animated sequence of representations, rather than showing a static node-link structure as in U.S. Patent 5,629,632, shows a dynamic node-link structure. The edits, however, serve primarily to add features representing new nodes along the outer perimeter of the representation as the representation makes the transition from the previous position to the destination position. As a result, the added features do not interfere with or reduce the perception of object constancy for features representing other elements.

An insert/delete event could result when the user requests expansion or contraction of a node or requests some other modification of the graph or the tree. An insert/delete event could also be received in the form of a call, and could thus provide a mechanism for automatic modification of the graph or tree without concurrent human control.

In response to an event of this type, client 212 can first make appropriate calls to routines in memory 214 to determine whether the requested modification of the graph or tree is acceptable, in box 330. For example, a technique for determining whether an expand signal is acceptable is described in relation to Fig. 7 of copending coassigned U.S. Patent Application ~~09/EEE,EEE~~ 09/124,338 (Attorney

Change(s) applied to document,
/J.L.B./
1/6/2012

In box 312, walker routines 222 could first perform any necessary layouts in the hyperbolic plane, and could also lay out any pending edits of the tree. Then, in box 314, walker routines 222 could map the tree into the unit disk, beginning with a starting node at a starting position, in the manner described in copending coassigned U.S. Patent Application 09/124,529 ~~09/CCC,CCC~~ (Attorney Docket No. D/98205Q2), entitled "Mapping a Node-Link Structure to a Rendering Space Beginning from any Node", incorporated herein by reference. For example, in response to a dragging event, the starting node could be the nearest node identified in box 310 and the starting position could be the next position along the path of motion. The starting node and starting position previously used for mapping could be used in response to an orientation or stretch event.

Change(s) applied to document, /J.L.B./ 1/6/2012

When the tree has been mapped, painter routines 224 can be called to paint the mapped tree in a display buffer, in box 316. During painting, painter routines 224 can mark new edits that occur in the tree as a result of node creation as described in copending coassigned U.S. Patent Application 09/124,338 ~~09/EEE,EEE~~ (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the Graph", incorporated herein by reference. Each edit can be marked by setting a flag or storing other appropriate data. When painting is completed, a swap of display buffers can be performed to present the tree as painted, thus providing a representation of the graph.

Node position data 232, which can be linked to or included within directed graph data structure 230, can include positions of nodes in a negatively curved space such as a hyperbolic plane and in a rendering space such as a two-dimensional unit disk. Node position data 232 can be accessed by routines in program memory 214.

The routines in program memory 214 can also access various miscellaneous data structures 234. Data structures 234 may, for example, include an extra data structure for mapping from a pair of node IDs to a link ID, implemented as a standard heap; this extra data structure allows lookup and insertion of a link ID in constant expected time.

C.2. Responding to Events

Fig. 5 shows how the system of Fig. 4 can respond to events by presenting representations of a graph.

In box 300, client 212 begins by obtaining a starting graph and by loading an initial set of elements into memory, such as through calls to create nodes as described in copending coassigned U.S. Patent Application ~~09/DDD,DDD~~ 09/124,474 (Attorney Docket No. D/98205Q3), entitled "Controlling Which Part of Data Defining a Node-Link Structure is in Memory", incorporated herein by reference. Expansion flags define a tree within the initial set of elements, as described in copending coassigned U.S. Patent Application ~~09/EEE,EEE~~ 09/124,338 (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within

Change(s) applied to document, /J.L.B./ 1/6/2012

information about a directed graph, which could be a combination of routines and data stored in memory 210 or could be independent of memory 210 as shown. For example, processor 202 could communicate with client 212 through a network.

The routines stored in program memory 214 can be grouped into several functions. Grapher routines 220 create and modify a data structure representing the directed graph defined by the information from client 212. Walker routines 222 respond to navigation signals and other user signals from keyboard 206 and mouse 208 by obtaining information from the directed graph data structure. Painter routines 224 provide signals to display 204 to cause it to present representations of the directed graph data structure. Math routines 226 can be called to obtain positions of elements of the directed graph in a layout space.

Data memory 216 in turn contains data structures accessed by processor 202 during execution of routines in program memory 214. Directed graph data structure 230, as noted above, can be created and modified by grapher routines 220 and can also be accessed by walker routines 222 and painter routines 224.

Further details about the implementation of directed graph data structure 230 are set forth in copending coassigned U.S. Patent Applications 09/124,474 ~~09/DDD,DDD~~ (Attorney Docket No. D/98205Q3), entitled "Controlling Which Part of Data Defining a Node-Link Structure is in Memory", and 09/124,338 ~~09/EEE,EEE~~ (Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the Graph", both incorporated herein by reference.

Change(s) applied to document, /J.L.B./ 1/6/2012